Tutorial AMPL

Parte IV

Eduardo Camponogara

Department of Automation and Systems Engineering
Federal University of Santa Catarina

October 2016

Summary

Knapsack Problem

$$\max \sum_{i=1}^{n} c_i x_i$$

$$\text{s.t.} : \sum_{i=1}^{n} a_i x_i \leq b$$

$$x_i \in \{0, 1\}$$

in which

- c_i is the value of item i;
- a_i is the weight of item i;
- b is the capacity of knapsack.

Cover

- A subset $C \subseteq \{1, \ldots, n\}$ of items such that:

$$\sum_{i \in C} a_i > b$$

is called cover inequality.

- The cover inequality

$$\sum_{i \in C} x_i \leq |C| - 1$$

is valid for the knapsack problem.

Cover

- A subset $C \subseteq \{1, \ldots, n\}$ of items such that:

$$\sum_{i \in C} a_i > b$$

is called cover inequality.

- The cover inequality

$$\sum_{i \in C} x_i \leq |C| - 1$$

is valid for the knapsack problem.

Minimum Cover

A cover C is a minimum cover if, for each $i \in C$,

$$\sum_{j \in C \setminus \{i\}} a_i \leq b$$

meaning that $C \setminus \{i\}$ is not a cover for all $i \in C$.

Summary

Knapsack Example

For the problem:

$$\max \quad 42x_1 + 30x_2 + 26x_3 + 16x_4 \tag{1a}$$
$$\text{s.t.} : 83x_1 + 61x_2 + 49x_3 + 20x_4 \leq 100 \tag{1b}$$
$$x_1, x_2, x_3, x_4 \in \{0, 1\} \tag{1c}$$

obtain all minimum covers and the respective cover inequalities.

Cover Inequalities

Cover sets:

$$C_1 = \{1, 2\}$$
$$C_2 = \{1, 3\}$$
$$C_3 = \{1, 4\}$$
$$C_4 = \{2, 3\}$$

AMPL Tasks

1. Implement the AMPL model for the knapscak problem (1).
2. Solve the continuous, linear relaxation of the knapsack problem.
3. Introduce the cover inequalities for $C_1, \ldots, C_4$ and solve the resulting linear relaxation.

AMPL Tasks

example8.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
set K = 1..4 by 1;
var x{k in K} binary;
param c{k in K};
param a{k in K};
param b;
# Part 2: Objective Function
maximize objective: sum{k in K} c[k]*x[k];
# Part 3: Constraints
subject to R1: sum{k in K} a[k]*x[k] <= b;
```

AMPL Tasks

example8.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
set K = 1..4 by 1;
var x{k in K} binary;
param c{k in K};
param a{k in K};
param b;
# Part 2: Objective Function
maximize objective: sum{k in K} c[k]*x[k];
# Part 3: Constraints
subject to R1: sum{k in K} a[k]*x[k] <= b;
```

AMPL Tasks

example8.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
set K = 1..4 by 1;
var x{k in K} binary;
param c{k in K};
param a{k in K};
param b;
# Part 2: Objective Function
maximize objective: sum{k in K} c[k]*x[k];
# Part 3: Constraints
subject to R1: sum{k in K} a[k]*x[k] <= b;
```

Summary

Cutting-Plane Algorithm with Gomory Cuts

Consider the IP problem below:

$$\begin{aligned} \max \;\; & 2.1x_1 + 2.3x_2 \\ \text{s.t.}: \;\; & -x_1 + 3x_2 \leqslant 11.5 \\ & x_1 + x_2 \leqslant 8 \\ & 2x_1 - x_2 \leqslant 10 \\ & x_1, x_2 \geq 0, \;\; \text{integer} \end{aligned}$$

Tasks:

1. Find the optimal solution using AMPL.
2. Perform 2 iterations of the Cutting-Plane Algorithm using Gomory Cuts. Solve each LP as we did in class.

Cutting-Plane Algorithm with Gomory Cuts

Consider the IP problem below:

$$\begin{aligned} \max \;\; & 2.1x_1 + 2.3x_2 \\ \text{s.t.}: \;\; & -x_1 + 3x_2 \leqslant 11.5 \\ & x_1 + x_2 \leqslant 8 \\ & 2x_1 - x_2 \leqslant 10 \\ & x_1, x_2 \geq 0, \;\; \text{integer} \end{aligned}$$

Tasks:

1. Find the optimal solution using AMPL.
2. Perform 2 iterations of the Cutting-Plane Algorithm using Gomory Cuts. Solve each LP as we did in class.

AMPL Task: Solving IP Problem

example9.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0, integer;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] <= 11.5;
subject to R2: x[1] + x[2] <= 8;
subject to R3: 2*x[1] - x[2] <= 10;
```

AMPL Task: Solving IP Problem

example9.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0, integer;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] <= 11.5;
subject to R2: x[1] + x[2] <= 8;
subject to R3: 2*x[1] - x[2] <= 10;
```

AMPL Task: Solving IP Problem

example9.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0, integer;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] <= 11.5;
subject to R2: x[1] + x[2] <= 8;
subject to R3: 2*x[1] - x[2] <= 10;
```

AMPL Task: Solving IP Problem

Defining the LP:

$$
\begin{aligned}
\max \quad & 2.1x_1 + 2.3x_2 \\
\text{s.t.}: \quad & -x_1 + 3x_2 + s_1 = 11.5 \\
& x_1 + x_2 + s_2 = 8 \\
& 2x_1 - x_2 + s_3 = 10 \\
& x_1, x_2, s_1, s_2, s_3 \geq 0
\end{aligned}
$$

AMPL Task: Applying Cutting-Plane Algorithm

example10.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0;
var s{1..3} >= 0;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] + s[1] = 11.5;
subject to R2: x[1] + x[2] + s[2] = 8;
subject to R3: 2*x[1] - x[2] + s[3] = 10;
```

AMPL Task: Applying Cutting-Plane Algorithm

example10.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0;
var s{1..3} >= 0;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] + s[1] = 11.5;
subject to R2: x[1] + x[2] + s[2] = 8;
subject to R3: 2*x[1] - x[2] + s[3] = 10;
```

AMPL Task: Applying Cutting-Plane Algorithm

example10.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0;
var s{1..3} >= 0;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] + s[1] = 11.5;
subject to R2: x[1] + x[2] + s[2] = 8;
subject to R3: 2*x[1] - x[2] + s[3] = 10;
```

Generate Gomory Cut

The dictionary of the LP is given below:

- Upper bound: 17.775.
- $B^{-1} * [B\ N]$ is:

$$\begin{bmatrix} 1 & 0 & 0 & -0.25 & 0.75 \\ 0 & 1 & 0 & 0.25 & 0.25 \\ 0 & 0 & 1 & 0.75 & -1.25 \end{bmatrix}$$

- $B^{-}1 * b$ gives (3.125, 4.875, 8.625).

Generate Gomory Cut

For a basic fractional row u, the Gomory Cut is given by:

$$\sum_{j \in NB} (\bar{a}_{uj} - \lfloor \bar{a}_{uj} \rfloor) x_j \geqslant \bar{a}_{uo} - \lfloor \bar{a}_{uo} \rfloor$$

Applying for the first row (x_1), we obtain:

$$(-0.25 - \lfloor -0.25 \rfloor)s_1 + (0.75 - \lfloor 0.75 \rfloor)s_2 \geq 3.125 - \lfloor 3.125 \rfloor$$

which produces the cutting-plane:

$$0.75s_1 + 0.75s_2 \geq 0.125$$

AMPL Task: Applying Cutting-Plane Algorithm

example10.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0;
var s{1..4} >= 0;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] + s[1] = 11.5;
subject to R2: x[1] + x[2] + s[2] = 8;
subject to R3: 2*x[1] - x[2] + s[3] = 10;
subject to R4: -0.75*s[1] - 0.75*s[2] + s[4] = -0.125;
```

AMPL Task: Applying Cutting-Plane Algorithm

example10.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0;
var s{1..4} >= 0;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] + s[1] = 11.5;
subject to R2: x[1] + x[2] + s[2] = 8;
subject to R3: 2*x[1] - x[2] + s[3] = 10;
subject to R4: -0.75*s[1] - 0.75*s[2] + s[4] = -0.125;
```

AMPL Task: Applying Cutting-Plane Algorithm

example10.mod:

```
# Part 1: Variable Declaration (var, set, param, etc)
var x{1..2} >= 0;
var s{1..4} >= 0;
# Part 2: Objective Function
maximize objective: 2.1*x[1] + 2.3*x[2];
# Part 3: Constraints
subject to R1: -x[1] +3*x[2] + s[1] = 11.5;
subject to R2: x[1] + x[2] + s[2] = 8;
subject to R3: 2*x[1] - x[2] + s[3] = 10;
subject to R4: -0.75*s[1] - 0.75*s[2] + s[4] = -0.125;
```

AMPL Tutorial

- Thank you for attending this lecture!!!